IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): Modelski	
Application No.: 09/741,857	Group Art Unit: 2452
Filed: 12/22/2000	Examiner: Truong
Title: Multi-Thread Packet Processor	Confirmation No.: 8573
Attorney Docket No.: 120-100	

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REPLY BRIEF

Sir:

Please enter this Reply Brief.

I. Real Party in Interest

The real party in interest is Nortel Networks Limited.

II. Related Appeals and Interferences

Appellants are not aware of any related appeals or interferences.

III. Status of the Claims

Claims 1-3 and 5-18 are pending in this application. All of the pending claims are rejected. Claims 1-3, 5-6, 11-15, and 17 are previously presented. Claims 7-10, 16 and 18 are original. The rejections of claims 1 and 5 are the subject of this appeal.

IV. Status of Amendments

All submitted amendments have been entered and considered.

V. Summary of Claimed Subject Matter

This invention generally relates to data processing architectures, and more particularly to a multi-thread packet processor for rapidly processing data packets. In the past, processor designers have been able to obtain throughput improvements by greater integration, by reducing the size of the circuits, and by the use of single-chip reduced instruction set computing processors. It is commonly understood, however, that physical size reductions cannot continue indefinitely, and there are limits to continually increasing processor clock speeds. The presently claimed invention enhances throughput with a multi-thread packet processor which processes data packets using a multi-threaded

pipelined machine such that delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline because no instruction depends on a preceding instruction because each instruction in the pipeline is executed for a different thread.

The limitations recited in claims 1 and 5 are supported by the specification and drawing as indicated below in **bold**.

1. (previously presented) A method for routing a data packet comprising:

producing a plurality of threads associated with the packet, each thread being a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads;

Each AM 42,56,70,84 is configured with a plurality of independent threads, for packet processing. The packet processing effected by an AM includes identifying the packets and determining what to do with them. AMs 42,56,70,84 do not modify packets. Each thread has a full context of data registers, address registers, program counter, special registers, and additional resources. The threads share a common integer pipeline and global register set. The state of each thread is independent from the state of all other threads. Page 11, lines 18-23.

assigning a thread identifier (TID) to each of the threads and maintaining an activity status for each thread;

Threads and their register content are identified by a Thread Identification (TID) number. Status is provided to indicate which threads are active or inactive, enabled or disabled, etc. Page 11, lines 23-25.

for each thread, selecting a pipeline from a plurality of pipelines, at least some of which are specialized, and forwarding that thread to the selected pipeline, such that processing of each packet is divided into multiple independent threads which are processed by multiple pipelines, and such that delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline.

In addition to the AM integer pipeline that starts the execution of every AM instruction, each AM has access to several specialized co-processor units such as EMEs 120,266, HE 158, etc. The TID follows the instruction everywhere in the AM or co-processor pipelines. The TID is also the primary mechanism of control between all co-processing units, packet data interfaces, packet pre-classifiers, and the integer pipeline. Page 11, line 25 through page 12, line 1. Each of AMs 42,56,70,84 is configured with an internal integer pipeline and shared access to several additional specialized processing pipelines. These specialized shared pipelines may be viewed as co-processors and include: 1) EMEs 120,166 - supports lookups, memory accesses and atomic arithmetic; 2) HE 158 - supports programmable 24-bit Cyclic Redundancy Checking (CRC) based hashes of 64-bit keys; 3) lMEs 122,162 - supports atomic arithmetic and memory accesses; CIF 160 - supports additional lookups, memory accesses and atomic arithmetic in shared CLUE memory; 4) BAP 10 - supports access of peripheral devices. Page 12, lines 5-12. The multi-thread packet processor processes data packets using a multi-threaded pipelined machine, wherein no instruction depends on a preceding

instruction because each instruction in the pipeline is executed for a different thread. Page 3, lines 6-9.

5. (previously presented) An apparatus for routing a packet comprising:

a memory for storing:

a plurality of threads associated with the packet, each thread being a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads;

Each AM 42,56,70,84 is configured with a plurality of independent threads, for packet processing. The packet processing effected by an AM includes identifying the packets and determining what to do with them. AMs 42,56,70,84 do not modify packets. Each thread has a full context of data registers, address registers, program counter, special registers, and additional resources. The threads share a common integer pipeline and global register set. The state of each thread is independent from the state of all other threads. Page 11, lines 18-23.

a unique Thread Identifier (TID) for each thread; and

an activity status for each thread; and

Threads and their register content are identified by a Thread Identification (TID) number. Status is provided to indicate which threads are active or inactive, enabled or disabled, etc. Page 11, lines 23-25.

an analysis machine including a plurality of pipelines, at least some of which are specialized, the analysis machine selecting a pipeline for each thread and forwarding that thread to the selected pipeline such that processing of each packet is divided into multiple

independent threads which are processed by multiple pipelines, and such that delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline.

In addition to the AM integer pipeline that starts the execution of every AM instruction, each AM has access to several specialized co-processor units such as EMEs 120,266, HE 158, etc. The TID follows the instruction everywhere in the AM or co-processor pipelines. The TID is also the primary mechanism of control between all co-processing units, packet data interfaces, packet pre-classifiers, and the integer pipeline. Page 11, line 25 through page 12, line 1. Each of AMs 42,56,70,84 is configured with an internal integer pipeline and shared access to several additional specialized processing pipelines. These specialized shared pipelines may be viewed as co-processors and include: 1) EMEs 120,166 - supports lookups, memory accesses and atomic arithmetic; 2) HE 158 - supports programmable 24-bit Cyclic Redundancy Checking (CRC) based hashes of 64-bit keys; 3) lMEs 122,162 - supports atomic arithmetic and memory accesses; CIF 160 - supports additional lookups, memory accesses and atomic arithmetic in shared CLUE memory; 4) BAP 10 - supports access of peripheral devices. Page 12, lines 5-12. The multi-thread packet processor processes data packets using a multi-threaded pipelined machine, wherein no instruction depends on a preceding instruction because each instruction in the pipeline is executed for a different thread. Page 3, lines 6-9.

VI. Grounds of Rejection to be Reviewed on Appeal

A. Claims 1 and 5 are rejected under 35 U.S.C. 103(a) based on US 2006/0156303 (Hooper) in view of US 7,093,109 (Davis), and further in view of US 7,111,156 (Mang).

B. Claims 1 and 5 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

VII. Argument

A. The cited combination of references fails to teach or suggest all the limitations recited in claims 1 and 5.

Three basic criteria must be met in order to establish a *prima facie* case of obviousness. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Third, the prior art references must teach or suggest all the claim limitations. (MPEP §2143) At a minimum, the cited combination fails to satisfy the third requirement.

"All words in a claim must be considered in judging the patentability of that claim against the prior art." *In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970). Claims 1 and 5 both recite that "delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline." This limitation is never mentioned in the office action and there is no evidence that it was considered. As stated in the Summary at page 3, lines 3-7, "methods and apparatuses consistent with the principles of the present invention, as embodied and broadly described herein, provide for a multi-thread packet processor which processes data packets using a multi-threaded pipelined machine, wherein no instruction depends on a preceding instruction because each instruction in the pipeline is executed for a different thread." It should therefore be apparent that the claim limitation quoted above is not only worthy of consideration, but is one of the primary features of the invention. Because the Examiner has not even alleged that the feature is taught in the cited references, and because the feature is not taught in the cited references, the prior art fails to teach all of the claim limitations.

The Examiner's Answer asserts that Hooper teaches that "delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline" at the abstract and paragraphs 0023, 0032-0035. Because none of the cited passages actually mentions the affect of one thread upon another thread, it is assumed that the examiner is asserting that the limitation is implied by Hooper's description of the threads as being "independent." However, the independence of threads does not

imply that delay of one thread does not affect another thread. As stated in the Background of this application at page 2, lines 5-10:

> Multi-thread processing divides a processing task into **independently** executable sequences of instructions called threads and the processor, recognizing when an instruction has caused it to be idle (i.e., first thread), switches from the instruction causing the memory latency to another instruction (i.e., second thread) **independent** from the former instruction. **At some point, the threads that had caused the processor to be idle will be ready and the processor will <u>return</u> to those threads**. (emphasis added)

Note that the delay of the first independent thread affects processing of the second independent thread. It should therefore be appreciated that thread independence does not imply a lack of relationship based on delay.

The Examiner concedes that Hooper fails to disclose producing a plurality of threads associated with the packet where each thread is a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads. However, the Examiner asserts that the feature is taught by Davis at column 3, lines 49-51 and column 4, lines 5-6, 9-11. The passage at column 3 states "the invention also involves the use of a prefetch buffer in connection with a plurality of independent thread processes in such a manner as to avoid an immediate stall when execution is granted to an idle thread." Note that the threads are not described as being independently executable in terms of delay interdependence, but merely "independent," i.e., separate. Consequently, the cited passage simply describes threads that are separate from one another, without regard for any quality of being executable

without regard to the state of other threads. The passage at column 4 states that "the invention specifically relates to independent processes in each of the instruction execution threads (each of which relates to a different packet being processed), and the invention specifically deals with *latency in accessing data*," and "each of the execution threads is an independent process executing a sequence of instructions as the threads are allowed to gain access to the processor hardware." Therefore, the prior art fails to teach all of the claim limitations.

The Examiner's Answer denies that the examiner conceded "that Hooper fails to disclose producing a plurality of threads associated with the packet where *each thread is a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads*," (emphasis added) as discussed above. As evidence the examiner cites the Office Action at pages 3 and 4 (with reference to claim 1). Appellant was referring to the examiner's statement at page 6, lines 1-3 of the Office Action (with reference to claim 5) that "Hooper does not explicitly disclose *each thread being a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads*." (emphasis added)

The Examiner concedes that Hooper and Davis fail to teach that at least some of the pipelines are specialized, but asserts that Mang teaches the feature at column 3, lines 63-67; column 4, lines 1-8; column 10, lines 32-37; and column 11, lines 1-14. The passage at column 3 states that "the computation engine 12, which is discussed in greater detail with reference to FIGS. 3 and 8-15 below, receives the ordered operation codes 48 and generates resultants 50

therefrom." Appellant can find no colorable argument that any pipeline is described in the cited passage, and certainly not a specialized pipeline. The passage at column 4 states:

> The objective of the arbitration module 14 is to order the operation codes 48 such that the computation engine 12 runs at capacity (i.e. the pipeline within the computation engine is always full and the resources in the computation engine are efficiently utilized). Thus, for every operation cycle of the computation engine 12, the arbitration module 14 attempts to provide it with an operation code for execution.

The passage at column 4 describes a pipeline, but there is no indication that it is specialized, so one would naturally assume that it is non-specialized, i.e., a general pipeline. The passage at column 10 states:

> The circuit 400 differs from that illustrated in FIG. 8 in that it includes a plurality of pre-accumulation registers 421-423. The selection block 430 selects a second operand for the adder 440 from a set of potential operands that includes the contents of the plurality of pre-accumulation registers 421-423, data in the accumulation buffer 460 and the memory 450, and additional operands 434.

Appellant can find no colorable argument that any pipeline is described in the cited passage, and certainly not a specialized pipeline. Finally, the passage at column 11 states:

> the stored data causes the operand selection information 432 to select the data in the appropriate pre-accumulation register using the selection block 430. Note the priority within each of the threads is only limited to priority with respect to other operation codes that utilize the pre-accumulation register for that thread. In other words, a first operation (AxB) may be executed where the result of this operation is stored in the pre-accumulation register corresponding to that particular thread that issued the operation code. That particular

> thread can then perform numerous other operations that do not utilize the pre-accumulation register prior to performing an operation that utilizes the stored result of (AxB) contained within its respective preaccumulation register.

Again, Appellant can find no colorable argument that any pipeline is described in the cited passage, and certainly not a specialized pipeline.

In view of the above it will be appreciated that the cited references fail to teach either specialized pipelines or producing a plurality of threads associated with the packet where each thread is a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads. Perhaps more importantly, the Examiner has not even considered the limitation that "delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline." Whether considered individually or as a whole, these limitations distinguish the cited references.

B. The term "specialized" in claims 1 and 5 is not indefinite because it is understood in the art and described in detail in the specification.

In reviewing a claim for compliance with 35 U.S.C. 112, second paragraph, the examiner must consider the claim as a whole to determine whether the claim apprises one of ordinary skill in the art of its scope and, therefore, serves the notice function required by 35 U.S.C. 112, second paragraph, by providing clear

warning to others as to what constitutes infringement of the patent. See, e.g., *Solomon v. Kimberly-Clark Corp.*, 216 F.3d 1372, 1379, 55 USPQ2d 1279, 1283 (Fed. Cir. 2000). Accordingly, a claim term that is not used or defined in the specification is not indefinite if the meaning of the claim term is discernible. *Bancorp Services, L.L.C. v. Hartford Life Ins. Co.*, 359 F.3d 1367, 1372, 69 USPQ2d 1996, 1999-2000 (Fed. Cir. 2004) During examination, the claims must be interpreted as broadly as their terms reasonably allow. *In re American Academy of Science Tech Center*, 367 F.3d 1359, 1369, 70 USPQ2d 1827, 1834 (Fed. Cir. 2004) (The USPTO uses a different standard for construing claims than that used by district courts; during examination the USPTO must give claims their broadest reasonable interpretation in light of the specification.). This means that the words of the claim must be given their plain meaning unless the plain meaning is inconsistent with the specification. *In re Zletz*, 893 F.2d 319, 321, 13 USPQ2d 1320, 1322 (Fed. Cir. 1989); *Chef America, Inc. v. Lamb-Weston, Inc.*, 358 F.3d 1371, 1372, 69 USPQ2d 1857 (Fed. Cir. 2004).

The ordinary meaning of the term "specialized" is "designed, trained or fitted for one particular purpose or occupation." Merriam-Webster Dictionary, 2007. The opposite of "specialized" is "unspecialized," meaning not specialized or modified for a particular purpose or function. Interpretting the specialized pipelines in view of the common meaning of the term "specialized," it is clear that the pipelines are not general, but rather designed for a particular purpose. Further, the ordinary meaning of the term is consistent with the description of the specialized pipelines in the specification at page 12, lines 5-12, which states that:

> Each of AMs 42,56,70,84 is configured with an internal integer pipeline and shared access to several additional specialized processing pipelines. These specialized shared pipelines may be viewed as co-processors and include: 1) EMEs 120,166 - supports lookups, memory accesses and atomic arithmetic; 2) HE 158 - supports programmable 24-bit Cyclic Redundancy Checking (CRC) based hashes of 64-bit keys; 3) lMEs 122,162 - supports atomic arithmetic and memory accesses; CIF 160 - supports additional lookups, memory accesses and atomic arithmetic in shared CLUE memory; 4) BAP 10 - supports access of peripheral devices.

Clearly, a specialized processing pipeline is simply a pipeline that performs a subset of the functions of a general purpose processing pipeline. Consequently, the claim language provides clear warning to others as to what constitutes infringement.

The Examiner's Answer counters that "nowhere in the citation discloses the use of designing a pipeline for a particular purpose as admitted by applicant." It is suggested that the passage from the specification at page 12, lines 5-12 quoted above describes such specialized pipelines, i.e., "1) EMEs 120,166 - supports lookups, memory accesses and atomic arithmetic; 2) HE 158 - supports programmable 24-bit Cyclic Redundancy Checking (CRC) based hashes of 64-bit keys; 3) lMEs 122,162 - supports atomic arithmetic and memory accesses; CIF 160 - supports additional lookups, memory accesses and atomic arithmetic in shared CLUE memory; 4) BAP 10 - supports access of peripheral devices."

The Examiner's Answer also states that "the specification [and dictionary] fails to provide a standard for ascertaining the requisite *degree* for the claim limitation." In other words, the examiner requires an indication of how much the

pipelines are specialized. Appellant suggests that the examiner is mistaken in suggesting that the term presents a question of degree. More particularly, a pipeline is either designed for one purpose or it is not – there are no *partially* specialized pipelines known in the art. As a practical matter this difference can be manifested as a special purpose coprocessor chip versus a general purpose microprocessor, and those of ordinary skill in the art are aware of the differences between those things.

VIII. Conclusion

The rejections are improper for at least the reasons set forth above. Appellants accordingly request that the rejections be reversed and the application put forward for allowance.

Respectfully submitted,

/Holmes W. Anderson/
Holmes W. Anderson
Reg. No. 37,272
Attorney for Assignee

Date: July 30, 2009

Anderson Gorecki & Manaras LLP
33 Nagog Park
Acton MA 01720
(978) 264-4001

Appendix A - Claims

1. (previously presented) A method for routing a data packet comprising:

producing a plurality of threads associated with the packet, each thread being a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads;

assigning a thread identifier (TID) to each of the threads and maintaining an activity status for each thread;

for each thread, selecting a pipeline from a plurality of pipelines, at least some of which are specialized, and forwarding that thread to the selected pipeline, such that processing of each packet is divided into multiple independent threads which are processed by multiple pipelines, and such that delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline.

2. (previously presented) The method according to claim 1, further comprising: transferring the first thread from an input buffer to a packet task manager; dispatching the first thread from the packet task manager to an analysis machine; classifying the first thread in the analysis machine; and modifying and forwarding the first thread in a packet manipulator.

3. (previously presented) The method according to claim 1, wherein the activity status indicates that a status of the associated thread is one of active, inactive or waiting.

4. (cancelled)

5. (previously presented) An apparatus for routing a packet comprising:

a memory for storing:

a plurality of threads associated with the packet, each thread being a sequence of instructions that facilitates packet routing and that is independently executable with respect to other ones of the threads;

a unique Thread Identifier (TID) for each thread; and

an activity status for each thread; and

an analysis machine including a plurality of pipelines, at least some of which are specialized, the analysis machine selecting a pipeline for each thread and forwarding that thread to the selected pipeline such that processing of each packet is divided into multiple independent threads which are processed by multiple pipelines, and such that delay in processing of a first packet routing thread in a first pipeline does not affect processing of a second packet routing thread in a second pipeline.

6. (previously presented) The apparatus according to claim 5, wherein one pipeline is dedicated to directly manipulating individual data bits of a bit field a packet task manager operationally connected to said analysis machine, and a packet manipulator operationally connected to said analysis machine.

7. (original) The apparatus according to claim 6, wherein said analysis machine is multi-threaded.

8. (original) The apparatus according to claim 6, wherein said analysis machine has 32 threads.

9. (original) The apparatus according to claim 6, further comprising: a packet task manager operationally connected to said analysis machine; a packet manipulator operationally connected to said analysis machine; and a global access bus including a master request bus and a slave request bus separated from each other and pipelined.

10. (original) The apparatus according to claim 6, further comprising: an external memory engine operationally connected to said analysis machine; and a hash engine operationally connected to said analysis machine.

11. (previously presented) The apparatus according to claim 9, further comprising: packet input global access bus program code , stored in a computer readable memory and operable when executed to control a flow of data packet information from a flexible input data buffer to the analysis machine.

12. (previously presented) The apparatus according to claim 9, further comprising: packet data global access bus program code, stored in a computer readable memory and operable when executed to control a flow of packet data between a flexible data input bus and the packet manipulator.

13. (previously presented) The apparatus according to claim 9, further comprising: statistics data global access bus software code used for connection of the analysis machine to the packet manipulator.

14. (previously presented) The apparatus according to claim 9, further comprising: private data global access bus software code used for connection of the analysis machine to an internal memory engine submodule.

15. (previously presented) The apparatus according to claim 9, further comprising: lookup global access bus software code used for connection of the analysis machine to an internal memory engine submodule.

16. (original) The apparatus according to claim 9, further comprising: results global access bus software code used for providing flexible access to an external memory.

17. (previously presented) The apparatus according to claim 5, wherein the activity status indicates that the associated multi-IP packet thread status is one of active, inactive or waiting.

18. (original) The apparatus according to claim 9, further comprising: a bi-directional access port operationally connected to said analysis machine; an input buffer operationally connected to said analysis machine; and an output buffer operationally connected to said analysis machine.

Appendix B - Evidence Submitted

None.

Appendix C - Related Proceedings

None.